AMENDMENTS TO THE SPECIFICATION

Please amend the fourth paragraph on page 7 as follows:

Figure 3 illustrates an expanded partial view of the "receive" side of switch logic 202 wherein it is shown that in one embodiment pointer interpreter 208 treats as "almost full" FIFO buffer 210 when it is **at least** twenty-seven (27) or less slots (where each slot is of size sufficient to hold at least one column of SPE data) away from being completely full.

Please amend the fifth paragraph on page 7 as follows:

Figure 4 shows an expanded partial view of the "transmit" side of switch logic 202 wherein it is shown that in one embodiment pointer interpreter 214 treats as "almost full" FIFO buffer **[[**210**]]** **216** when it is three (3) or less slots (where each slot is of size sufficient to hold at least one column of SPE data) away from being completely ~~empty~~ **full**; also shown is that in one embodiment "almost full" is set as five (5) slots rather ~~then~~ **than** three (3), for reasons of practicality.

Please amend the paragraph beginning on line 10 of page 9 and ending on line 23 of page 9 as follows:

It has been discovered by the inventors that since pointer interpreter 208 and pointer generator 212 are keyed to different frame structures (e.g., standard SONET frame 108 and non-standard SONET frame 204 respectively), what constitutes an "almost full" or "almost empty" FIFO buffer 210 will vary dependent upon whether FIFO buffer 210 is being viewed from the standpoint of pointer interpreter 208 or pointer generator 212. For example, since pointer interpreter 208 tends to cooperate in writing 87 column "chunks" of payload data (e.g., the number of columns between overhead columns) to FIFO buffer 210, what constitutes "almost empty" and "almost full" from the standpoint of pointer interpreter 208 will be different from that seen by pointer generator **[[**202**]]** 212, which tends to read out the entire payload contents from FIFO buffer 210 en masse subsequent to the construction of the 27 byte overhead data structure of non-standard SONET frame 204.

Please amend the last paragraph on page 9 as follows:

Conversely, insofar as pointer generator 212 is concerned, there could be instances in which pointer interpreter 208 is reading three columns of overhead data and not writing data to FIFO buffer 210. Consequently, in order to be safe, pointer generator 212 preferably should treat the "almost empty" condition as being ~~3 or less~~ **at least three** columns of data away from a completely empty FIFO buffer 210 (which for an STS-1 frame equates to 3 bytes)**, i.e., should the buffer contain 3 or fewer columns of data it would be considered "almost empty."**[[;]] ~~however~~ **However**, in at least one implementation it has been empirically determined that due to practical considerations it is useful to have the "almost empty" indicator set to 5 columns (which for an STS-1 frame equates to 5 bytes)**,i.e., should the buffer contain 5 or fewer columns of data it would be considered "almost empty."**

Please amend the paragraph beginning on line 16 of page 11 and ending on line 25 of page 11 as follows:

Conversely, insofar as pointer interpreter 214 is concerned, there could be instances in which pointer ~~interpreter~~ **generator** **[[**208**]]** 218 is constructing three columns of overhead data and not reading data from FIFO buffer 216. Consequently, in order to be safe, pointer interpreter 214 preferably should treat the "almost full" condition as being 3 or less columns of data away from a completely ~~empty~~ **full** FIFO buffer 216 (which for an STS-1 frame equates to 3 bytes); however, in at least one implementation it has been empirically determined that due to practical considerations it is useful to have the "almost empty" indicator set to 5 columns, which for an STS-1 frame equates to 5 bytes.